Electronic Colloquium on Computational Complexity, Report No. 9 (2012)

Computational Complexity, NP Completeness and Optimization Duality: A Survey

Prabhu Manyem
Department of Mathematics
Shanghai University
Shanghai 200444, China.
Email: prabhu.manyem@gmail.com

Julien Ugon
School of Science, Info. Tech. and Engr.
University of Ballarat
Mt Helen, VIC 3350, Australia.
Email: j.ugon@ballarat.edu.au

November 28, 2011

Abstract

We survey research that studies the connection between the computational complexity of optimization problems on the one hand, and the duality gap between the primal and dual optimization problems on the other. To our knowledge, this is the first survey that connects the two very important areas. We further look at a similar phenomenon in finite model theory relating to complexity and optimization.

Keywords. Computational complexity, Optimization, Mathematical programming, Duality, Duality Gap, Finite model theory.

AMS Classification. 90C25, 90C26, 90C46, 49N15, 65K05, 68Q15, 68Q17, 68Q19 and 68Q25.

1 Introduction

In optimization problems, the **duality gap** is the difference between the optimal solution values of the primal problem and the dual problem. The relationship between the duality gap and the computational complexity of optimization problems has been implicitly studied for the last few decades. The connection between the two phenomena has been subtly acknowledged. The *gap* has been exploited to design good approximation algorithms for NP-hard optimization problems [2, 13, 24]. However, we have been unable to locate a single piece of literature that addresses this issue explicitly.

This report is an attempt to bring a great deal of evidence together and specifically address this issue. Does the existence of polynomial time algorithms for the primal and the dual problems mean that the duality gap is zero? Conversely, does the existence of a duality gap imply that either the primal problem or the dual problem is (or both are) NP-hard? Is there an inherent connection between computational complexity and *strong duality* (that is, zero duality gap)?

Vecten and Fasbender (independently) were the first to discover the optimization duality [11]. They observed this phenomenon in the *Fermat-Torricelli problem*: Given a triangle T_1, find

ISSN 1433-8092

the equilateral triangle circumscribed outside T_1 with the maximum height. They showed that this *maximum* height H is equal to the *minimum* sum of the distances from the vertices to T_1 to the *Torricelli point*[1]. Thus, this problem enjoys strong duality.

The apparent connection between the duality gap and computational complexity was considered more than thirty years ago. Linear Programming (LP) is a well known optimization problem. In the mid 1970s, before Khachiyan published his ellipsoid algorithm, LP was thought to be polynomially solvable, precisely because it obeys *strong duality*; that is, the duality gap is zero. For a good description of the ellipsoid algorithm, the reader is referred to the good book by Fang and Puthenpura [10]. Strong duality also places the decision version of Linear Programming in the class NP $\cap$ CoNP; see Lemma 12 below.

We should stress that this is **not** a survey on Lagrangian duality or any other form of optimization duality. Rather, this is a survey on the **connections** and **relationships** between the computational complexity of optimization problems and duality.

2 Definitions

A few definitions are provided in this section.

Definition 1. *[15] A **P-optimization** problem Q is a tuple $Q = \{I_{\mathrm{Q}}, F_{\mathrm{Q}}, f_{\mathrm{Q}}, opt_{\mathrm{Q}}\}$, where*

(i) I_{Q} *is a set of instances to* Q,

(ii) $F_{\mathrm{Q}}(I)$ *is the set of feasible solutions to instance* I,

(iii) $f_{\mathrm{Q}}(I, S)$ *is the* objective function *value to a solution* $S \in F_{\mathrm{Q}}(I)$ *of an instance* $I \in I_{\mathrm{Q}}$. *It is a function* $f : \bigcup_{I \in I_{\mathrm{Q}}} [\{I\} \times F_{\mathrm{Q}}(I)] \to \mathbb{R}_0^+$ *(non-negative reals)*[2], *computable in time polynomial in the size* $|A|$ *of the domain* A *of* I[3],

(iv) For an instance $I \in I_{\mathrm{Q}}$, $opt_{\mathrm{Q}}(I)$ *is either the minimum or maximum possible value that can be obtained for the objective function, taken over all feasible solutions in* $F_{\mathrm{Q}}(I)$.

$opt_{\mathrm{Q}}(I) = \max_{S \in F_{\mathrm{Q}}(I)} f_{\mathrm{Q}}(I, S)$ *(for P-maximization problems),*

$opt_{\mathrm{Q}}(I) = \min_{S \in F_{\mathrm{Q}}(I)} f_{\mathrm{Q}}(I, S)$ *(for P-minimization problems),*

(v) The following decision problem is in the class **P***: Given an instance* I *and a non-negative constant* K*, is there a feasible solution* $S \in F_{\mathrm{Q}}(I)$*, such that* $f_{\mathrm{Q}}(I, S) \geq K$ *(for a P-maximization problem), or* $f_{\mathrm{Q}}(I, S) \leq K$ *(in the case of a P-minimization problem)?*

And finally,

[1] The Torricelli point X is indeed the one with the least sum of the distances $|AX| + |BX| + |CX|$ from the vertices A, B and C of T_1.

[2] Of course, when it comes to computer representation, rational numbers will be used.

[3] Strictly speaking, we should use $|I|$ here, where $|I|$ is the length of the representation of I. However, $|I|$ is polynomial in $|A|$, hence we can use $|A|$.

(vi) *An optimal solution $S_{\text{opt}}(I)$ for a given instance I can be computed in time polynomial in $|I|$, where $opt_{\text{Q}}(I) = f_{\text{Q}}(I, S_{\text{opt}}(I))$.*

The set of all such $\mathbf{P}$*-optimization problems is the* $\mathbf{P_{opt}}$ *class.* ∎

A similar definition, for NP-optimization problems, appeared in Panconesi and Ranjan (1993) [20]:

Definition 2. *An **NP-optimization** problem is defined as follows. Points (i)-(iv) in Def. 1 above apply to NP-optimization problems, whereas (vi) does not. Point (v) is modified as follows:*

(v) The following decision problem is in $\mathbf{NP}$*: Given an instance I and a non-negative constant K, is there a feasible solution $S \in F_{\text{Q}}(I)$, such that $f_{\text{Q}}(I, S) \geq K$ (for an NP-maximization problem), or $f_{\text{Q}}(I, S) \leq K$ (in the case of an NP-minimization problem) ?*

The set of all such $\mathbf{NP}$*-optimization problems is the* $\mathbf{NP_{opt}}$ *class, and* $\mathbf{P_{opt}} \subseteq \mathbf{NP_{opt}}$. ∎

In this paper, an *optimization problem* will refer to an *NP-optimization* problem. Furthermore, we only work with a particular class of optimization problems called *mathematical programming problems* defined in Def. 3, where the set of feasible solutions $F_{\text{Q}}(I)$ to a given instance I is defined by the set of *constraints*.

Definition 3. *(Mathematical programming problem $\mathcal{P}$)*

Given*:*

Objective function: a function $f(\mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^n$ is a vector of variables; ($\mathbf{x}$ is called a solution to $\mathcal{P}$);

Constraints:

1. *An m_1 (≥ 0) number of constraints $\mathbf{g}(\mathbf{x}) = \mathbf{b}$, where $\mathbf{b} \in \mathbb{R}^{m_1}$ is a vector of constants, $\mathbf{g}$ is a set of functions $\{g_i(\mathbf{x}),\ 1 \leq i \leq m_1\}$; and*

2. *an m_2 (≥ 0) number of constraints $\mathbf{h}(\mathbf{x}) \geq \mathbf{c}$, where $\mathbf{c} \in \mathbb{R}^{m_2}$ is another vector of constants and $\mathbf{h}$ is a set of functions $\{h_j(\mathbf{x}),\ 1 \leq j \leq m_2\}$.*

The functions f, g_i and h_j are computable in time polynomial in the input size of $\mathbf{x}$. As inputs to Turing machines, f, g_i and h_j are encoded in bit strings in such as way that for a given $\mathbf{x}$, they are computable in time polynomial in the size of the encoding of $\mathbf{x}$. The quantities n, m_1 and m_2 are non-negative integers and part of the input.

To Do*:*

Determine a solution $\mathbf{x}^*$ such that

$f(\mathbf{x}^*) = \text{opt}_{\mathbf{x}}\ \{f(\mathbf{x} :\ \mathbf{g}(\mathbf{x}) = \mathbf{b}$ *and* $\mathbf{h}(\mathbf{x}) \geq \mathbf{c}\}$, *where*

$\text{opt} = \max\ (\min)$, *if $\mathcal{P}$ is a maximization (minimization) problem, respectively.*

Such an $\mathbf{x}^*$ (if it exists) is called an optimal solution to $\mathcal{P}$. ∎

Definition 4. *($\mathsf{D_1(r)}$: Decision problem corresponding to $\mathcal{P}$ in Def. 3)*

Given. *As in Def. 3. In addition, we are given a parameter $r \in \mathbb{R}$.*

To Do. *Determine if the set $\mathcal{F} = \emptyset$, where*
$\mathcal{F} = \{\mathbf{x} : \ \mathbf{g}(\mathbf{x}) = \mathbf{b}, \ \mathbf{h}(\mathbf{x}) \geq \mathbf{c}$ and $f(\mathbf{x}) \geq r\}$, if $\mathcal{P}$ is a maximization problem, and

$\mathcal{F} = \{\mathbf{x} : \ \mathbf{g}(\mathbf{x}) = \mathbf{b}, \ \mathbf{h}(\mathbf{x}) \geq \mathbf{c}$ and $f(\mathbf{x}) \leq r\}$, if $\mathcal{P}$ is a minimization problem. ■

$\mathcal{F}$ is the set of feasible solutions to the decision problem.

Remark 5. *A word of caution: For decision problems, the term "feasibility" includes the constraint on the objective function; if the objective function constraint is violated, the problem becomes infeasible.*

Definition 6. *[3] (Lagrangian Dual)*
Suppose we are given a minimization problem P_1 such as

$$\begin{aligned} \textit{Minimize} \quad & f(\mathbf{x}) : X \rightarrow \mathbb{R} \quad (X \subseteq \mathbb{R}^n), \\ \textit{subject to} \quad & \mathbf{g}(\mathbf{x}) = \mathbf{b}, \quad \mathbf{h}(\mathbf{x}) \geq \mathbf{c}, \\ \textit{where} \quad & \mathbf{x} \in \mathbb{R}^n, \ \mathbf{b} \in \mathbb{R}^{m_1} \ \textit{and}\ \mathbf{c} \in \mathbb{R}^{m_2}. \end{aligned} \tag{1}$$

*We call this the **primal** problem, whose feasible region is X. Assume that $\boldsymbol{b} = [b_1 \ b_2 \ \cdots \ b_{m_1}]^T$ and $\boldsymbol{c} = [c_1 \ c_2 \ \cdots \ c_{m_2}]^T$ are column vectors. Let $\mathbf{u} \in \mathbb{R}^{m_1}$ and $\mathbf{v} \in \mathbb{R}^{m_2}$ be two vectors of variables with $\mathbf{v} \geq \mathbf{0}$.*

*For a given primal problem as in P_1, the Lagrangian **dual** problem P_2 is defined as follows:*

$$\begin{aligned} \textit{Maximize} \quad & \theta(\mathbf{u}, \mathbf{v}) \\ \textit{subject to} \quad & \mathbf{v} \geq \mathbf{0}, \textit{where} \\ \theta(\mathbf{u}, \mathbf{v}) = \quad & \inf_{\mathbf{x} \in \mathbb{R}^n} \ \{f(\mathbf{x}) + \sum_{i=1}^{m_1} u_i(g_i(\mathbf{x}) - b_i) + \sum_{j=1}^{m_2} v_j(h_j(\mathbf{x}) - c_i)\}. \end{aligned} \tag{2}$$

*The **duality gap** is defined as $|f(\mathbf{x}^*) - \theta(\mathbf{u}^*, \mathbf{v}^*)|$, where $f(\mathbf{x}^*)$ $(\theta(\mathbf{u}^*, \mathbf{v}^*))$ is the optimal solution value to the primal (dual) problem, respectively.* ■

Note that $g_i(\mathbf{x}) - b_i = 0$ $[h_j(\mathbf{x}) - c_j \geq 0]$ is the i^{th} equality $[j^{th}$ inequality] constraint.

3 Background: Duality and the classes NP and CoNP

We now turn our attention to the relationship between the duality of an optimization problem, and membership in the complexity classes NP and CoNP of the corresponding set of decision problems. Decision problems are those with yes/no answers, as opposed to optimization problems that return an optimal solution (if a feasible solution exists).

Corresponding to P_1 defined above in (1), there is a set D_1 of decision problems, defined as $D_1 = \{D_1(r) \mid r \in \mathbb{R}\}$. The definition of $D_1(r)$ was provided in Def. 4.

Let us now define the computational classes NP, CoNP and P. For more details, the interested reader is referred to either [2] or [21]. We begin with the following well known definition:

Definition 7. ***NP*** *(respectively* ***P****) is the class of decision problems for which there exist non-deterministic (respectively deterministic) Turing machines which provide Yes/No answers in time that is polynomial in the size of the input instance. In particular, for problems in P and NP, if the answer is* ***yes****, the Turing machine (TM) is able to provide an "evidence" (in technical terms called a* certificate*), such as a feasible solution to a given instance.*

The class ***CoNP*** *of decision problems is similar to NP, except for one key difference: the TM is able to provide a certificate only for* ***no*** *answers.*

From the above, it follows that for an instance of a problem in NP $\cap$ CoNP, the corresponding Turing machine can provide a certificate for both yes and no instances. ■

For example, if $D_1(r)$ in Def. 4 above is in NP, the certificate will be a feasible solution; that is, an $\mathbf{x} \in \mathbb{R}^n$ which obeys the constraints

$$\mathbf{g}(\mathbf{x}) = \mathbf{b}, \ \mathbf{h}(\mathbf{x}) \leq \mathbf{c} \text{ and } f(\mathbf{x}) \geq r. \tag{3}$$

On the other hand, if $D_1(r) \in$ CoNP, the certificate will be an $\mathbf{x} \in \mathbb{R}^n$ that violates at least one of the $m_1 + m_2 + 1$ constraints in (3).

Remark 8. *For problems in NP, for Yes instances, extracting a solution from the certificate is not always an efficient (polynomial time) task. Similarly, in the case of CoNP, pinpointing a violation from a Turing machine certificate* [4] *is not guaranteed to be efficient either.*

Remark 9. *$P \subseteq NP$, because any computation that can be carried out by a deterministic TM can also be carried out by a non-deterministic TM. The problems in P are decidable deterministically in polynomial time.*

The class P is the same as its complement Co-P. That is, P is closed under complementation.

Furthermore, Co-P ($\equiv$ P) is a subset of CoNP. We know that P is a subset of NP. Hence $P \subseteq NP \cap CoNP$. Thus for an instance of a problem in P, the corresponding Turing machine can provide a certificate for both yes and no instances. ■

We are now ready to define what is meant by a *tight dual*, and how it relates to the intersection class of problems, NP $\cap$ CoNP. Note that for two problems to be tight duals, it is sufficient if they are tight with respect to just one type of duality (such as Lagrangian duality, for example). However in this paper, we have only considered Lagrangian duality, and hence we restrict our attention to this type.

Definition 10. *Tight duals and the class TD.*
Consider two optimization problems P_a and P_b, as defined in Def. 3. Given a primal problem, let its dual be defined as in Def. 6. Then P_a and P_b are ***dual to each other*** *if the dual of one problem is the other.*

Suppose P_a and P_b are dual to each other, with zero duality gap; then we say that P_a and P_b are ***tight duals****. For any $r \in \mathbb{R}$, let $D_a(r)$ and $D_b(r)$ be the decision versions of P_a and P_b respectively.*

Let ***TD*** *be the class of all decision problems whose optimization versions have tight duals. That is,* ***TD*** *is the set of all problems $D_a(r)$ and $D_b(r)$ for any $r \in \mathbb{R}$.* ■

[4] We thank WenXun Xing and PingKe Li of Tsinghua University (Beijing) for ponting out the above.

Remark 11. *A word of caution. Tight duality is not the same as strong duality. For strong duality, it is sufficient if there exist feasible solutions to the primal and the dual such that the duality gap is zero. For tight duality to hold, we also require that the primal problem P_a and the dual problem P_b be dual to one another.*

One way in which duality gaps are related to the classes NP and CoNP is as follows:

Lemma 12. *[21] TD $\subseteq$ NP $\cap$ CoNP.*

From Remark 9 and Lemma 12, we know that both TD and P are subsets of NP $\cap$ CoNP. But is there a containment relationship between TD and P? That is, is either TD $\subseteq$ P or P $\subseteq$ TD? This is the subject of further study in this paper, with particular reference to Lagrangian duality.

Remark 13. *We should mention that in several cases, given a primal problem P, even if we are able to find a dual problem D such that the dual of P is D, it does not necessarily follow that the dual of D is P. That is,the dual of the dual need not be the primal. P and D are not necessarily duals of each other. We do not include such (P, D) pairs in **TD**. Among primal-dual pairs of problems, TD is a restricted class. Convex problems satisfying Slater's condition and their dual belong to the class TD.*

4 Lack of Strong Duality results in NP hardness

In this section, we will review results from the literature, which show that the lack of strong duality imply that the optimization problem in question is NP-hard, assuming that the primal problem obeys the *constraint qualification* assumption as stated below in Def. 17. Here we work with Lagrangian duality. Results for other types of duality such as Fenchel, geometric and canonical dualities require further investigation.

Let us define what we mean by weak duality (as opposed to tight duality and strong duality):

Definition 14. *Given a primal problem P_1 and a dual problem P_2, as defined in Def. 6, the pair (P_1, P_2) is said to obey **weak duality** if $\theta(\mathbf{u}, \mathbf{v}) \leq f(\mathbf{x})$, for every feasible solution $\boldsymbol{x}$ to the primal and every feasible solution ($\boldsymbol{u}$, $\boldsymbol{v}$) to the dual.*

Definition 15. *If, in addition to the assumptions from Def. (14), there exist a primal feasible solution $\bar{\mathbf{x}}$ and a dual feasible solution $(\bar{\mathbf{u}}, \bar{\mathbf{v}})$ such that equality holds, that is: $f(\bar{\mathbf{x}}) = \theta(\bar{\mathbf{u}}, \bar{\mathbf{v}})$, then the pair (P_1, P_2) is said to be obey **strong duality**.*

The following theorem from [3] guarantees that the feasible solutions to Lagrangian dual problems (1) and (2) indeed obey *weak duality*:

Theorem 16. *If $\boldsymbol{x}$ is a feasible solution to the primal problem in (1) and ($\boldsymbol{u}$, $\boldsymbol{v}$) is a feasible solution to the dual problem in (2), then $f(\mathbf{x}) \geq \theta(\mathbf{u}, \mathbf{v})$.*

We shall now define a special type of convex program, called a *convex program with constraint qualification*, which is one with an assumption about the existence of a feasible solution in the interior of the domain.

Definition 17. Convex program (convex optimization problem).
Given. *A convex set* $X \subset \mathbb{R}^n$*, two convex functions* $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ *and* $\mathbf{g}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^{m_1}$*, as well as an affine function* $\mathbf{h}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^{m_2}$.

To do. *Minimize* $f(\mathbf{x})$*, subject to* $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$*,* $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ *and* $\mathbf{x} \in X$.

Definition 18. Convex program with constraint qualification.
Same as the optimization problem in Def. 17, except that we include the following constraint qualification assumption *(known as* Slater's condition*): There is an* $\mathbf{x_0} \in X$ *such that* $\mathbf{g}(\mathbf{x_0}) < \mathbf{0}$ *and* $\mathbf{h}(\mathbf{x_0}) = \mathbf{0}$.

(*Note*: Of course, the functions above can be written in the same form as in Def. 1 and 2. In such a case, we can define $(\mathbf{g}(\mathbf{x}) - \mathbf{b})$ to be a convex function and $(\mathbf{h}(\mathbf{x}) - \mathbf{c})$ to be an affine function, where $\mathbf{b} \in \mathbb{R}^{m_1}$ and $\mathbf{c} \in \mathbb{R}^{m_2}$.)

Definition 19. *In Def. 17, if any of the* $(m+1)$ *functions* $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ *and* $\mathbf{g}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^{m_1}$ *is not convex, then the optimization problem is said to* ***non-convex****.*

For the remainder of this section, we will assume primal constraint qualification; that is, we assume that constraint qualification is applied to the primal optimization problem. The following theorem provides sufficient conditions under which strong duality can occur:

Theorem 20. ***Strong Duality*** *[3]. If (i) the primal problem is given as in Def. 17, and (ii) the primal and dual problems have feasible solutions, then the primal and dual optimal solution values are equal (that is, the duality gap is zero):*

$$\begin{aligned} &\inf\{f(\mathbf{x}) : \mathbf{x} \in X,\ \mathbf{g}(\mathbf{x}) \leq \mathbf{0},\ \mathbf{h}(\mathbf{x}) = \mathbf{0}\} = \sup\{\theta(\mathbf{u}, \mathbf{v}) : \mathbf{v} \geq \mathbf{0}\}, \\ &\theta(\mathbf{u}, \mathbf{v}) = \inf_{\mathbf{x} \in X}\{f(\mathbf{x}) + \sum_{i=1}^{m_1} u_i g_i(\mathbf{x}) + \sum_{j=1}^{m_2} v_j h_j(\mathbf{x})\}, \end{aligned} \tag{4}$$

where $\theta(\mathbf{u}, \mathbf{v})$ *is the dual objective function.* ■

Using the contrapositive statement of Theorem 20, we get the following result:

Corollary 21. *(to Theorem 20) If there exists a duality gap using Lagrangian duals, then either the primal or the dual is not a convex optimization problem. (Remember, we are assuming constraint qualification.)*

The *Subset Sum* problem is defined as follows: Given a set S of positive integers $\{d_1, d_2, \cdots, d_k\}$ and another positive integer d_0, is there a subset P of S, such that the sum of the integers in P equals d_0?

Using a polynomial time reduction from the *Subset Sum* problem to a non-convex optimization problem (see Def. 19), Murty and Kabadi (1987) showed the following:

Theorem 22. *[18] If an optimization problem is non-convex, it is NP-hard.*

In certain cases, non-convex problems have an equivalent convex formulation, for example, through strong duality. Such a dual transformation, where a convex problem B is a dual of a non-convex problem A such that the duality gap between them is zero, is called hidden convexity [4, 5]. In such cases, the reformulated convex problem is also NP-hard; otherwise the primal non-convex problem can be solved efficiently, thus violating Theorem 22.

Definition 23. *Standard Quadratic Program (SQP) [6]. Minimize the function* $\mathbf{x}^T Q\mathbf{x}$*, where* $\mathbf{x} \in \Delta$ *(the standard simplex in* $\mathbb{R}^n_+$*). The* n *vertices of* Δ *are at a unit distance (in the positive direction) along each of the* n *axes of* $\mathbb{R}^n$*.* Q *is a given symmetric matrix in* $\mathbb{R}^{n\times n}$*.*

The converse of Theorem 22 is not true. A convex optimization problem in general is NP-hard, SQP being an example. SQP is non-convex; however in [6], Bomze and de Clerk prove that it has an exact convex reformulation as a *copositive programming* problem. SQP is known to be NP-hard, since its decision version contains the max-clique problem in graphs as a special case. From this, it follows that copositive programming is also NP-hard; see [8] for more on this topic.

From Corollary 21 and Theorem 22, it follows that

Theorem 24. *Assuming constraint qualification, if there exists a duality gap using Lagrangian duals, then either the primal or the dual is NP-hard.*

These results are true for Lagrangian duality. For other types of duality such as Fenchel, geometric and canonical dualities, this requires further investigation.

5 Does Strong Duality Imply Polynomial Time Solvability?

At this time, such a proof (of whether a duality gap of zero implies polynomial time solvability of the primal and the dual problems) appears possible only for very simple problems, since estimating the duality gap appears extremely challenging for many problems.

It is not known whether strong duality results in polynomial time solvability in general. The only class of problems that we know where strong duality and polynomial time solvability (using interior point methods) occur together is Linear Programming. See Nesterov and Nemirovski [19] for more on the theory of interior point methods.

There are primal-dual problem pairs where both problems are convex and NP-hard, even though they enjoy strong duality under certain weak conditions such as Slater's condition [7].

More investigation is needed to answer the question *Does Strong Duality Imply Polynomial Time Solvability?* in a general setting. We conjecture the following:

Conjecture 25. *(Strong duality is a necessary but not sufficient condition.)*
If two optimization problems P_a *and* P_b *are polynomially solvable and one of them is the dual of the other, then they exhibit strong duality. However, the converse of this need not be true; that is, strong duality need not imply polynomial solvability.*

Consider quadratic programming [5] problems with a single quadratic constraint (QCQP) [25]. The primal problem P_0 is given by:

$$\begin{aligned} \text{Minimize} \quad & P(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{f}^T\mathbf{x} \\ \text{subject to} \quad & \frac{1}{2}\mathbf{x}^T B\mathbf{x} \leq \mu, \end{aligned} \tag{5}$$

[5]Thanks to Shu-Cherng Fang (NCSU, USA) for his input here.

where A and B are non-zero $n \times n$ symmetric matrices, $\mathbf{f} \in \mathbf{R}^n$, and $\mu \in \mathbf{R}$. (A, B, $\mathbf{f}$ and μ are given.)

Strong duality for a variation of QCQP was established by Moré and Sorensen in 1983 [17] (as described in [1]); they call theirs as the Trust Region Problem (TRP). The problem is to minimize a quadratic function q_0, subject to a norm constraint $\mathbf{x}^T\mathbf{x} \leq \delta^2$. This is an example of a non-convex problem where strong duality holds. Rendl and Wolkowicz [23] show that the TRP can be reformulated as an unconstrained concave maximization problem,

The observations above are for quadratic programming problems with a single quadratic constraint. It would be interesting to see what happens for quadratic programming problems with two constraints

Semidefinite Programming (SDP). Ramana [22] exhibited strong duality for the SDP problem. However, the complexity of SDP is unknown; it was shown in [22] that the decision version of SDP is NP-complete if and only if NP = CoNP.

[Note: The existence of algorithms polynomial *in the number of iterations* (see [19]) seems to be often assumed to mean that SDP, or indeed convex problems in general, are polynomial time solvable. This is NOT correct, as the above result in [22] shows.]

5.1 Solving optimization problems using a decision Turing machine

In a recent paper [16], we have demonstrated an additional computational benefit arising from strong duality. Primal-dual problem pairs that are in the class NP and obey strong duality can be solved by a single call to a *decision Turing machine*, that is, a Turing machine that provides a Yes/No answer (if the answer is yes, then it can provide a feasible solution which supports the Yes answer). Previously, it was only known that optimization problems require multiple calls to a decision Turing machine (for example, doing a binary search on the solution value to obtain an optimal solution). For more details, the reader is referred to [16].

6 Descriptive Complexity and Fixed Points

On a final note, we would like to briefly describe a similar phenomenon which occurs in the field of Descriptive Complexity, which is the application of Finite Model theory to computational complexity. In particular, we would like to mention least fixed point (LFP) computation. A full description would be beyond the scope of this paper. However, we would like to briefly mention a few related concepts and phenomena.

For a good description of least fixed points (LFP) in existential second order (ESO) logic, the reader is referred to [12] (chapters 2 and 3) and [9]. If the input structures are ordered, then expressions in LFP logic can describe polynomial time (PTIME) computation [12].

The input instance to an LFP computation consists of a *structure* $\mathbf{A}$, which includes a domain set A and a set of (first order) relations R_i, each with arity r_i, $1 \leq i \leq J$. The LFP computation works by a stagewise addition of tuples from A, to a new relation P (of some arity k). If P_i represents the relation (set of tuples) after stage i, then $P_i \subseteq P_{i+1}$. The transition from P_i to P_{i+1} is through an operator Φ, such that $P_{i+1} = \Phi(P_i)$. At the beginning, P is empty, that is, $P_0 = \emptyset$. For some value of i, say when $i = f$, if $P_f = P_{f+1}$, a *fixed point* has been reached.

Without going into details, let us just say that such a fixed point, reached as above, is also a *least* fixed point (LFP) if the operator Φ can be chosen in a particular manner. The interested reader is referred to [12] (chapter 2) for details.

Note that the number of elements in P can be at most $|A|^k$ (where $|A|$ is the number of elements in A), which is polynomial in the size of the domain. Hence $f \leq |A|^k$, so an LFP is achieved within a polynomial number of stages.

Similar to LFP, we can also define a *greatest* fixed point (GFP). This is obtained by doing the reverse; we start with the entire set A^k of k-ary tuples from the universe A, and then removing tuples from P in stages. At the beginning, $P_0 = A^k$. In further stages, $P_i \supset P_{i+1}$. The GFP is reached at stage g if $P_g = P_{g+1}$.

The logic that includes LFP and GFP expressions is known as **LFP logic**. It expresses decision problems (those with a Yes/No answer), such as those in Def. 4 and 7. To be feasible, a solution should also obey the objective function constraint ($f(\mathbf{x}) \geq K$ or $f(\mathbf{x}) \leq K$).

The LFP computation expresses decision problems based on maximization. Before the fixed point is reached, the solution is infeasible; that is, the number of tuples in the fixed point relation P is insufficient. However, once the fixed point is reached, the solution becomes feasible. Similarly, the GFP computation expresses decision problems based on minimization.

Problem. An interesting problem arising in LFP logic is this: For what type of primal-dual optimization problem pairs will the LFP and GFP computation meet at the same fixed point? Does this mean that such a pair is polynomially solvable?

7 Conclusion and Further Study

Let us again stress that this is **not** a survey on Lagrangian duality or any other form of optimization duality. Rather, this is a survey on the **connections** and **relationships** between the computational complexity of optimization problems and duality.

In this paper, we have touched the tip of the iceberg on a very interesting problem, that of connecting the computational hardness of an optimization problem with its duality characteristics. A lot more study is required in this area.

Another issue is that of *saddle point* for Lagrangian duals. This is a decidable problem; we can do brute force and find the primal and dual optimal solutions; this will tell us if there is a duality gap. If the gap is zero, then there is a saddle point.

(Jeroslow [14] showed that the integer programming problem with quadratic constraints is undecidable if the number of variables is unbounded, which is an extreme condition. However, if each variable has a finite upper and lower bound, then the number of solutions is finite and thus it is possible to determine the best solution in finite time.)

However, this problem would be NP-complete, unless we can tell whether it has a saddle point by looking at the structure of the problem or by running a polynomial time algorithm.

We hope that this paper will motivate further research in this very interesting topic.

8 Acknowledgments

The first author was supported by (i) a visiting fellowship from the National Cheng Kung University (NCKU), Tainan, Taiwan; (ii) the Shanghai Leading Academic Discipline Project #S30104; and (iii) the National Natural Science Foundation of China (# 11071158). Support from all sources is acknowledged.

We also thank Wen-Xun Xing and Ping-Ke Li (Tsinghua University, Beijing), Shu-Cherng Fang (North Carolina State University, USA) and Ruey-Lin Sheu (National Cheng-Kung University, Taiwan) for their valuable comments.

References

[1] K. Anstreicher and H. Wolkowicz. On Lagrangian relaxation of quadratic matrix constraints. *SIAM Journal on Matrix Analysis and Applications*, 22(1), 41–55 (electronic) (2000).

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela and M. Protasi. *Combinatorial Optimization Problems and Their Approximability Properties*. Springer, Germany (1999).

[3] M.S. Bazaraa, H.D. Sherali and C.M. Shetty. *Non-Linear Programming: Theory and Algorithms (third edition)*. Wiley Interscience (2006).

[4] A. Beck. Convexity Properties Associated with Nonconvex Quadratic Matrix Functions and Applications to Quadratic Programming. *Journal of Optimization Theory and Applications*, 142, 1–29 (2009).

[5] A. Ben-Tal and M. Teboulle. Hidden convexity in some nonconvex quadratically constrained quadratic programming. *Mathemalical Programming*, 72, 51–63 (1996).

[6] I.M. Bomze and E. de Klerk. Solving Standard Quadratic Optimization Problems Via Linear, Semidefinite and Copositive Programming. *Journal of Global Optimization*, 24, 163–185 (2002).

[7] I.M. Bomze, M. Dür, E. de Klerk, C. Roos, A.J. Quist and T. Terlaky. On Copositive Programming and Standard Quadratic Optimization Problems. *Journal of Global Optimization*, 18, 301–320 (2000).

[8] M. Dür. *Copositive Programming - a Survey*. in M. Diehl, F. Glineur, E. Jarlebring and W. Michiels (eds.): *Recent Advances in Optimization and its Applications in Engineering*, pp. 3–20, Springer (2010).

[9] H.D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer (1999).

[10] S.C. Fang and S. Puthenpura. *Linear Optimization and Extensions: Theory and Algorithms*. Prentice Hall, New Jersey (1993).

[11] F. Giannessi. On the theory of Lagrangian duality. *Optimization Letters*, 1, 9–20 (2007).

[12] E. Grädel, P.G. Kolaitis, L. Libkin, M. Marx, J. Spencer, M.Y. Vardi, Y. Venema, and S. Weinstein. *Finite Model Theory and Its Applications.* Texts in Theoretical Computer Science. Springer (2007).

[13] D. Hochbaum (ed.). *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company, Boston, MA (1997).

[14] R.G. Jeroslow. There Cannot be any Algorithm for Integer Programming with Quadratic Constraints. *Operations Research*, 21, 221–224 (1973).

[15] P. Manyem. Syntactic Characterizations of Polynomial Time Optimization Classes. *Chicago Journal of Theoretical Computer Science*, 3, 2008.

[16] P. Manyem. Syntactic expressions to express NP-hard optimization problems and problems with zero duality gap, 2011. To appear in the journal *Optimization.*. Available at `http://arxiv.org/abs/0904.4331`.

[17] J.J. Moré and D.C. Sorensen. Computing a trust region step. *SIAM Journal on Scientific and Statistical Computing*, 4(3), 553–572 (1983).

[18] K.G. Murty and S.N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39, 117–129 (1987).

[19] Y. Nesterov and A. Nemirovskii. *Interior Point Polynomial Methods in Convex Programming: Theory and Applications.* Society for Industrial and Applied Mathematics, Philadelphia (1994).

[20] A. Panconesi and D. Ranjan. Quantifiers and approximation. *Theoretical Computer Science*, 107:145–163, 1993.

[21] C.H. Papadimitriou. *Computational Complexity.* Addison-Wesley, Reading, Massachusetts (1994).

[22] M.V. Ramana. An Exact Duality Theory for Semidefinite Programming and Its Complexity Implications. *Mathematical Programming*, 77, 129–162 (1997).

[23] F. Rendl and H. Wolkowicz. A semidefinite framework for trust region subproblems with applications to large scale minimization. *Mathematical Programming*, 77(2, Series B), 273–299 (1997).

[24] V.V. Vazirani. *Approximation Algorithms.* Springer, Germany (2001).

[25] W. Xing, S-C. Fang, D.Y. Gao, R-L. Sheu and L. Zhang. Canonical Dual Solutions to the Quadratic Programming over a Quadratic Constraint. Technical report, Department of Mathematical Sciences, Tsinghua University, Beijing, China (2008).

ECCC ISSN 1433-8092

http://eccc.hpi-web.de